ASHESI UNIVERSITY

NILM- BASED SYSTEM FOR ENERGY MONITORING OF COMPOUND HOUSES IN GHANA

CAPSTONE PROJECT

B.Sc. Electrical and Electronic Engineering

Nana Akua Agyemang Sereboo

2020

ASHESI UNIVERSITY

NILM- BASED SYSTEM FOR ENERGY MONITORING OF COMPOUND HOUSES IN GHANA

CAPSTONE PROJECT

Capstone Project submitted to the Department of Engineering, Ashesi University in partial fulfilment of the requirements for the award of Bachelor of Science degree in Electrical and Electronics Engineering.

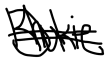
Nana Akua Agyemang Sereboo

2020

Declaration

I hereby declare that this capstone is the result of my own original work and that no part of it has been presented for another degree in this university or elsewhere.

Candidate's Signature:



……

Candidate's Name:

Nana Akua Agyemang Sereboo

……

Date:

29/05/2020

……

I hereby declare that preparation and presentation of this capstone were supervised in accordance with the guidelines on supervision of capstone laid down by Ashesi University.

Supervisor's Signature:

……

Supervisor's Name:

……

Date:

……

Acknowledgements

This project would not have been possible without the technical guidance from my supervisor, Francis Gatsi. I would also like to thank my guardians for the financial support and Ashesi University for their research facilities. Special thanks are extended to David Sasu and Amet Asamoah- Okyere for their insightful advice on solution formulation and data analysis. I am also grateful to the three tenants in compound houses who openly shared information with me during interviews.

Abstract

Compound housing is a housing arrangement that stemmed from cultural family living settings. Over the years, it has evolved to denote a group of different households that share a number of utilities including electricity. Electricity bills and payments are a concern in such living units due to the inability to split bills fairly. Previous work done to solve this are either expensive, or do not take into consideration the complexity of electricity connections in such houses. This thesis aims at verifying if the Feedforward Artificial Neural Network (ANN) can be used to split bills by implementing a Non-Intrusive Load Monitoring System for Compound Houses in Ghana. The proposed algorithm's performance is compared to Hidden Markov Model (HMM) and its variants. It was discovered that the Feedforward ANN had higher overall performance than the HMM and its variants, and is efficient for splitting electricity bills in compound houses.

Contents

Chapter 1: Introduction

1.1 Electricity in Ghanaian Homes

Electricity is the heartbeat of every economy. It is the cleanest and most efficient energy source that is used in the powering of machines for manufacturing, construction, and telecommunications. Sectors such as banking, hospitality, food, and entertainment all require some form of electrical energy to produce goods or render services. This goes to say electricity is a valuable resource that impacts life.

Electrical energy also makes life comfortable on the domestic level. Homes require electricity for basic day- to- day activities that have been made more efficient by technology. Washing machines, blenders, televisions, and water heaters are all devices that run on electricity to provide comfort and ease for humans.

Electricity reaches households through a complex network of switchgear and conductors. In Ghana, electricity is produced mainly by thermal plants (61% of total power generated), hydroelectric dams (produces 33% of total capacity) and Solar, which is less than a percent [1]**.** These various power generation plants generate up to 4,132MW of electrical power. The power is transmitted at high voltages to substations where the power is stepped down in stages to the nominal voltage of 240V. Households receive this power via service main lines. This power first runs through meters before reaching the household's distribution board. These meters track amount of electricity consumed in kilowatt-hours, and bills the customer at $0.06 per kilowatt- hour [2].

This metering method poses problems for certain accommodation arrangements in Ghana, particularly for compound houses. Compound houses are living units that stemmed from the culture of Ghanaians, which was to live with the extended family[3]. Compound houses are the most common housing arrangement in Ghana. According to Ghana

Statistical Services, 51.5% of Ghanaians live in such houses [4]. Currently, most compound houses are occupied by more than two different families. Limited shared spaces in such houses vary. They include shared bathrooms, kitchens, lounges, water meters and electric meters. In such living conditions, tensions rise because many people sharing a relatively small space usually feel an invasion of their privacy. These tensions also increase the difficulty in tenants sharing responsibility when it comes to utility bills payment.

Unfortunately, compound houses are not fully considered in electricity distribution. Most tenants of these compound houses constantly quarrel due to payment of electricity bills. Because there is only one meter that records amount of electricity consumed, each household is usually required to pay a fixed fraction of the total bill, regardless of the amount of electricity the individuals consume. Tenants usually refuse to pay bills because they believe others consumed more than they did.

Not only is this metering method a problem for tenants and landlords, it is also a problem for utility providers, Electricity Company of Ghana (E.C.G.) / Power Distribution Services (P.D.S). For homes with the traditional meter, which have to be manually read, utilities staff physically go to cut off electricity supply to compound houses that have not paid. In the process of doing this, they usually get abused by tenants when altercations as to who ought to pay occur.

Numerous approaches have been attempted in solving the issue, including buying separate meters. This method is expensive and requires that some portions of the household is rewired. Another method attempted focuses on a cheaper submeter that does not require rewiring to work [5]. Such methods assume that each room has separate wiring, which is not the case in many compound houses. Lighting circuits and sockets are usually looped to one another from the distribution board. As such, there is no solution provided for housing units without separate wiring from the distribution board.

1.2 Problem Definition

Tenants of compound houses that share one meter cannot adequately divide electricity bills for payment. This usually results in electricity bills not being paid, heated arguments and further, illegal connections to avoid paying electricity bills altogether. Wiring of rooms in compound houses are usually not separated. In other words, some rooms have looped circuits, making it difficult to separate points in the electricity network in the home.

1.3 Proposed Solution and Scope of Work

The project proposes Non-Intrusive Load Monitoring (NILM) as the method of separating the electricity consumption of different households in a compound house. This is a solution that would not require rewiring of the household and would work for compound houses where the rooms are not separately wired. Applying NILM to this problem would require that:

A. Each device in the household be identified by its electrical signature.
B. Aggregated electrical signal of the home is analysed to identify the appliances that are being used in real time based on their electrical signature.
C. Grouping of electrical signatures based on owners and calculation of power consumption of owner.
D. Information on amount of power consumed is delivered to different household units.

The scope of the project tackles the identification of appliances used in a typical compound house based on their power consumption. That is, this project would focus on identifying appliances based on their electrical signature using the Feedforward ANN algorithm, and compare this algorithm's performance to another commonly used NILM algorithm, namely Hidden Markov Model (HMM) and its variants.

1.4 Objectives of Project

This project seeks to

a. Design a hardware system that measures commonly used parameters in NILM, namely current and Apparent power.
b. Implement Feedforward ANN compatible with ATMEGA 328P.
c. Compare Feedforward ANN to HMM and its variants, namely Factorial Hidden Markov Model (FHMM), Additive FHMM and Hierarchical FHMM (HieFHMM)

Chapter 2: Literature Review

Related work and scientific papers in the knowledge area are discussed in this chapter to draw insights on design decisions and requirements, as well as provide relevant information to adapt the proposed method to solving the problem of electricity bill- splitting in compound houses. Literature review also ensures that this capstone highlights and addresses gaps in the knowledge area of splitting electricity bills in compound houses.

Most compound houses have complex electrical wiring. This is the case in many compound houses, as explained by [5]. In the author's capstone work, Andivi mentions that the joint wiring of rooms in compound houses makes it difficult to monitor how much each tenant consumes. The author utilized a current transducer, energy sensor, switch mode power converter, GSM communication module and a microcontroller to pick up electricity consumed, calculates electricity consumption, and sends an SMS to the tenants as to how much to pay. His suggested method of obtaining separate electricity bills for compound houses was made based on the assumption that each room in the compound house is wired separately, which is hardly the case in Ghana. This project seeks to provide a way to determine electrical consumption of tenants irrespective of the wiring of the compound house.

Another attempt to solve the issue of metering electricity consumption in compound houses is simply providing separate meters for different rooms or apartments in the compound house. Though it solves the issue of conflicts in the houses, it still is problematic, especially because tenants are unwilling to pay for shared spaces where electricity is used, such as corridors and "outside lighting". Aside that, it increases electricity consumption, which counters the nation's efforts to conserve electricity. In a research done in the Northern Area of Ghana covering Yendi, Tamale and Sawla, it is seen that electricity bills rose when separate meters were provided to tenants in compound houses [6]. This could be attributed to the

services charges on individual meters, and the change in consumer habits when a meter is purchased. Separate meters would also require special wiring done by professional electricians, which may come at an extra cost.

Due to the implications of separate metering, there is a need to implement a cost-effective method of monitoring electricity consumption in compound houses. There is also a need for a solution that works regardless of the wiring done in the home.

This project explores implementing Non-Intrusive Load Monitoring as a model for obtaining electrical consumption per sub household in a compound house.

Non-Intrusive Load Monitoring (NILM), synonymous with energy disaggregation, is a computational technique that estimates the power consumption of individual appliances from the total power recorded from a meter. Unlike Intrusive Load Monitoring, NILM is non-invasive, meaning that it does not require any changes to be made to electrical wiring and this makes it more favorable for energy monitoring in compound houses without separately wired homes. NILM is also selected over the Intrusive Load Monitoring approach because NILM requires only one main energy metering point [7]. This makes NILM less hardware-dependent than the Intrusive Load Monitoring approach, which usually requires an energy meter for each plugged device in the home.

NILM is a method used mainly for load shedding in smart grids, monitoring surveillance in homes, and general energy monitoring for energy conservation [8]. A more specific method of application of NILM is the Non-Intrusive Appliance Load Monitoring (NIALM) that was a concept birthed in the 1980's by Hart [9]. He matched the real and reactive power measurement of appliances to already stored values for each appliance. Other researchers built on the technology, by using other parameters to disaggregate total energy signal obtained from a single point. Some of these improvements were [10] that used load

transient shapes for event detection. Other works make use of artificial intelligence. And more recent work in NIALM focus on the current and voltage waveforms obtained at higher frequency sampling [11]. [12] works based on the assumption that the household has a smart meter and receives its measurements directly from it. No matter the method implemented, NILM algorithm has four main sections: data collection, event detection, feature extraction and load identification [13].

Data collection is carried out either by a smart meter or parameter-measuring sensors. Most compound houses in Ghana are not equipped with smart meters. As such, data collection in such scenarios would have to implement hardware using parameter-measuring sensors. The parameters usually measured are real power, reactive power, current, voltage, power factor and frequency. Time factor is important in measuring these parameters, since it is their behavior over time that would be used as features in the system.

Event detection analyzes the significant changes in the measured parameters. These parameters are categorized either as steady or transient [14]. The change in the steady or transient parameters of appliances indicate a change in state of the appliance. It may be speed adjustment to the device, function changes or simply a device turned on and off [15]. Feature sets are obtained from event detection, which determines what parameters are best for disaggregation. The most employed feature is apparent power [13].

Finally, load identification is carried out using the features described in the feature extraction stage. Recent implementations of load identification fall within the machine learning algorithms, which are classified as either supervised or unsupervised. Supervised techniques use offline training to design the classifier. Examples include Artificial Neural Networks (ANN), Deep Neural Networks and Linear-Chain Conditional random fields. Unsupervised techniques do not require initial training and can detect appliances without prior knowledge of

expected results. Examples of such algorithms are the Hidden Markov Method (HMM) and its variants. Aside these, there is also the use of Graph Signal Processing (GSP) in disaggregation. [16] presents the use of GSP on a graph filtering as well as on data. The most common techniques used in NILM are the ANN and the HMM. Both have their advantages and disadvantages. While HMM does not require rigorous training, it is time consuming and very complex. It is also difficult to distinguish appliances with similar power consumption[17]. This may be problematic as some devices in the compound house may have similar power consumption.

Literature reviewed in this chapter affirm that an NILM -based power monitor with a robust load identification algorithm for Ghanaian compound houses would be the best option for splitting electricity bills.

Chapter 3: Requirements

3.1 User Requirements

In order to undertake a project with a realistic and relevant deliverable, there is a need to undertake extensive research to first, confirm that the problem is indeed a problem. It is also necessary to empathize with people directly affected by the problem to ensure that this project contributes relevant knowledge and advances research in the topic area.

As such, interviews were conducted within October 2019 to gather information on the problem. In total, three interviews were conducted within Accra. The first in Nungua Kantamanto, the second in Ashaiman and the final one in Baatsona. All interviewees lived in compound houses and shared prepaid meters with other tenants. These prepaid meters were provided by the Electricity Company of Ghana (E.C.G) and are the sole source of electricity to each compound house.

Per the information gathered from the interviews (please refer to questions in appendix), shared metering is problematic for tenants of compound houses. All three sets of tenants explained how they tried different methods of sharing electricity bills, including splitting it evenly, taking turns to pay each month, having it factored into their rent, and splitting the bill based on general assessment of electrical devices in the house. Complaints expressed during the interviews included the fact that none of these arrangements had been effective. Tenants felt cheated, especially in instances when they were hardly at home. Others complained that other tenants always avoided paying their share of the bill, and that led to a lot of disputes.

When asked what they themselves were trying to do to solve the issues, one tenant mentioned that she and her family recently purchased their own meter, which was over a thousand Ghana Cedis (GHS1000.00). They admitted that their electricity bill was relatively higher now, but at least they had their peace of mind. Another tenant admitted to footing the entire bill whenever other tenants delay. All interviewees agreed that a better method of electricity billing is needed to reduce the conflicts that occur in their homes.

User requirements of this project emanate from the interviews that were conducted. Feedback received from the interviews conducted revealed that one of the methods used to share bills amongst tenants is the number of appliances each tenant has in their household. The bill is split based on the tenants' evaluation of how much power is consumed by each appliance in the compound house. That is to say, if one person owned a "large" appliance, such as a washing machine or electric oven, such a person is assumed to be consuming more power than others. This of course, may not be correct. Some tenants complained that inasmuch as they owned heavy appliances, they did not use them all the time. And even if they did, the true electricity consumption of devices in use could not be verified by anyone. The tenants asserted that whatever the solution is, it should be able to give each tenant an idea of how much they owe at every point in time. They suggested this because the prepaid system is employed in their homes, and the duration for recharge varies. They expressed the fact that if the solution would require a new meter, they would be highly reluctant to request for or buy one. There was also mention of the reluctance to pay the equivalent of a new meter for the solution to be implemented.

3.2 System Requirements

For the proposed approach, the deliverable should

- use a non-intrusive method to read electricity values from a main utility entry point. It should accurately obtain the aggregated current waveform of the household in real time.
- be able to disaggregate the various appliances from main power coming into the compound house. It should be able to do this by extracting the steady-state apparent power features of appliances.
- be able to detect the ON state and OFF state of each appliance.
- Have performance criteria better than HMM and its variants. That is:

$$F1\ score\ of\ proposed\ Feed\ Forward\ ANN\ > 0.854$$

Chapter 4: Design and Implementation

Requirements for this project are that the deliverable should be non-invasive and better performing than HMM implemented algorithms. NILM requires both hardware and software. As such, design decisions were made based on requirements and some assumptions. To implement the proposed solution, a low-cost energy monitor needs to be implemented to capture electrical parameters such as current and apparent power. These parameters need to be passed through NILM algorithm that would be implemented in software. Based on the results of the NILM, information on appliances in use are displayed.

4.1 Assumptions

In this project, the focus is to identify the devices that are on at any point in the compound house. Below, hardware design and software design of the system are discussed based on the assumptions that:

a. No tenant in the household uses the same brand and type of appliance.
b. The typical compound house is single phase.
c. The phase voltage to the compound house is always 240V.
d. All devices are electrical signature of Type I. They can either be only on or off, and do not have multiples states.

4.2 Hardware Design and Implementation

Hardware for an energy monitor is designed primarily to be compatible with Arduino coding. Each component was selected based on its compatibility with the Arduino, its cost, and

its ease of use. Components selected are the SCT-013-000 current transformer, ATMEGA 328P microcontroller, 9V rechargeable battery and an I2c LCD display.

4.2.1 Current Transformer

The primary and most essential component for the energy monitor is the current transformer. A current transformer is a device that scales usually large current to smaller current that is easy to measure safely [18]. This is the means by which electrical signal from the meter would be obtained for analysis by the system. To meet the requirements stated, the selected current transformer should be able measure up to 100A of current, since this is the maximum amount of current permitted in residential homes [19]. The selected current transformer should be non-invasive, meaning that it should require no rewiring or permanent wiring to the electrical network of the compound house. The current transformer should also be compatible with available microcontrollers. It must be cheap and must be easy to use. With these criteria, the ideal current transformer is the SCT-013-000. It was selected because

- It measures up to 100A of current
- It is non-invasive
- It is relatively cheap ($5.22)
- It is compatible with ATMEGA chips, esp8266

4.2.2 Microcontroller

The main criteria for selecting a microcontroller is its compatibility with the selected current transformer, its cost, ease of use, and availability. The most available microcontroller compatible is the ATMEGA 328P.

4.2.3 Power Source

The power source for the system is a 9V rechargeable battery. The system to be built cannot be dependent on the mains coming into the house. This is because if electricity is to be cut off from the house, the system must still be able to work.

4.2.4 Display

An OLED display was selected because of its ease of use and availability. Its main function is to display the ratio of tenants' electricity consumption.

4.2.5 Hardware Circuitry

Figure 4.1 is a schematic of the circuitry for the proposed system. The current transformer outputs current. The ATMEGA328P requires a voltage input between 0V and the reference voltage. As such, the current transformer is connected to a voltage divider that offsets the negative-positive output of the current transformer to oscillate with 2.5V as its midpoint. Any value of resistors of the same kind can be used for the voltage divider. It is ideal to select higher values of resistance to reduce energy consumption and save battery life.

The SCT 013-000 does not have an inbuilt burden resistor. A burden resistor converts current to a limited voltage. It is placed in the circuit to prevent the possibility of infinite voltage at the secondary side of the current transformer. Selection of a burden resistor is important because not only does it limit voltage, but it also improves its signal-to-noise ratio and bandwidth of the measure [20].

The procedure for selecting 33 Ω is listed below

- The maximum current the system would be measuring is 100A. This is used to calculate the maximum peak current.

  $Max\ primary\ peak\ current = 100A * \sqrt{2}$

  $Max\ primary\ peak\ current = 141.4214\ A$

- The peak current in the secondary side is calculated using the number of turns in the current transformer (100A : 0.05A= 2000).

  $Max\ Sec.Peak\ Current = \frac{Max\ primary\ peak\ current}{number\ \ of\ turns}$ **Eqn 4.1**

  $Max\ Secondary\ Peak\ Current = \frac{141.4214\ A}{2000}$

  $Max\ Secondary\ Peak\ Current = 0.0707\ A$

- The voltage across the burden resistor at the secondary peak current is half of the reference voltage for the ATMEGA328P chip (5V), for appropriate measurement resolution.
- Using Ohm's Law, resistance is calculated by

  $Burden\ Resistor = (\frac{Half\ ReferenceVoltage}{Secondary\ Peak\ Voltage})$ **Eqn 4.2**

  $Burden\ Resistor = \left(\frac{2.5\ V}{0.0707\ A}\right)$

  $Burden\ Resistor = 35.3553\ \Omega$

  The next available standard resistor is 33 **Ω**. A lower value was selected so that the voltage generated would not be higher than the desired voltage [21].

The power source is a rechargeable 9V battery that supplies 5V to the system. The voltage regulator LM7805 is connected to the power source as stated in its datasheet [22].

The ATMEGA328P is boot-loaded in order to use Arduino IDE to code the microcontroller. As such, a resonator, some capacitors and resistors as seen in the diagram are connected to the microcontroller [23].

OLED is connected as specified in Table 4.1. The OLED runs on 5V [24] and can be powered from the output of the power source.

Table 4.1 OLED connection to ATMEGA328P

Pin on OLED	Pin on ATMEGA 328P	Description
VCC	5V	Power supply
GND	GND	Ground
SCL	27	Clock Lines
SDA	28	Data Lines

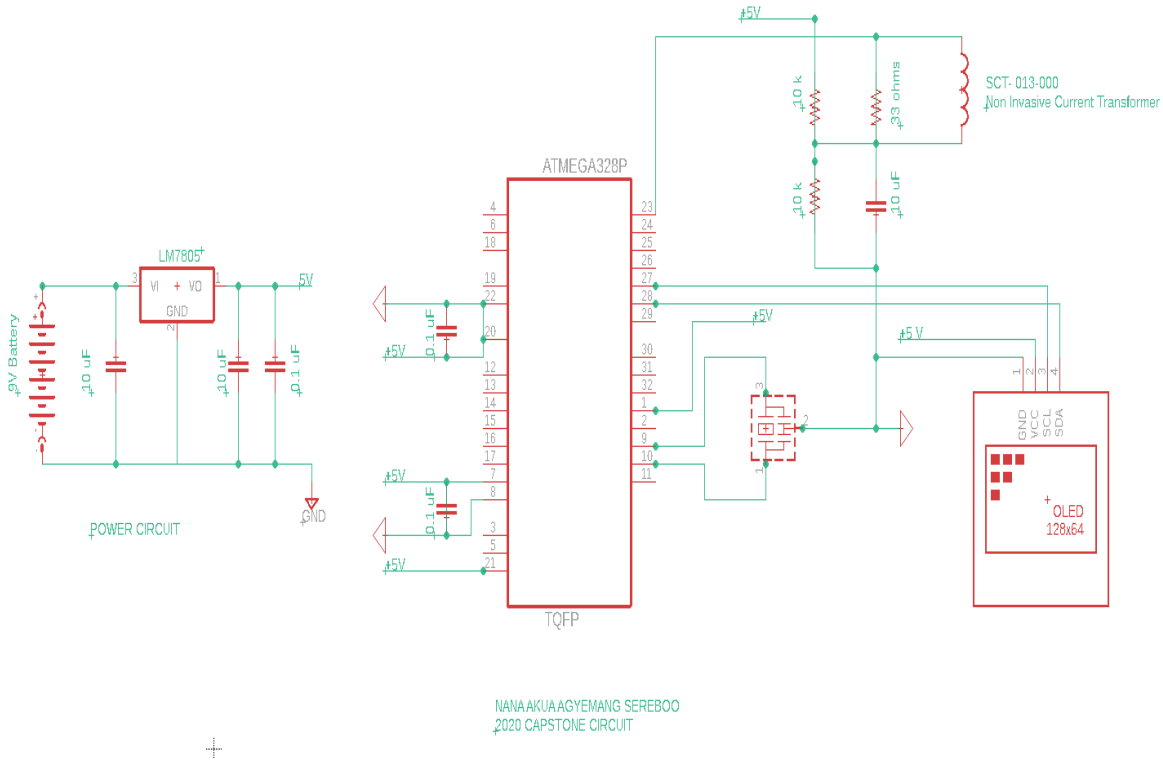


Figure 4.1. Schematic of hardware design of NILM- based energy monitor

4.3 Software Design and Implementation

The most frequently used algorithms for NILM are Artificial Neural Networks and Hidden Markov Models and their variants. This design uses the Feedforward ANN for its load identification. This is chosen over Hidden Markov Model and its variants because it requires less sampled data and is more robust for use in future work, in which more features and different state devices need to be added to the system [17]. The flow diagram of the software design is shown in Fig 4.2.

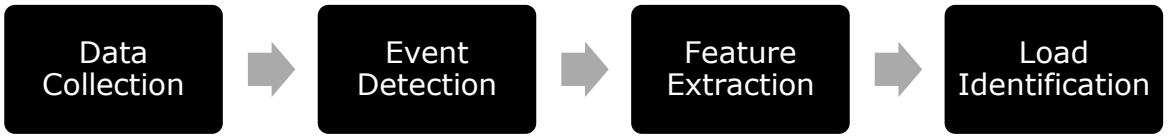


Fig 4.2 Basic flow diagram of Feedforward NILM software

4.3.1 Data Collection

Dataset from the Reference Energy Disaggregation Data Set (REDD) house one in the low frequency power dataset sampled at three seconds was obtained from [25]. Dataset online was used in order to have a uniform comparison of HMMs by using standard datasets. It was also due to the COVID-19 pandemic, which hindered the acquiring of hardware components to implement measurement circuit. The data obtained was then organized into csv format, and all channels appended into one file. Out of the total of ten appliances, five were selected: refrigerator, lighting, microwave, electric heater, and stove.

4.3.2 Event Detection

Expert heuristics was used in determining the change in states of the appliances. A python script (code attached in appendix) was used to determine on and off states of devices using the total power change metric.

4.3.3 Feature Extraction

The data set was sampled at three seconds [26]. As such, it is possible to use parameters such as current, voltage, and apparent power as features for generating feature sets. This dataset provides only apparent power (S) and that is adequate for implementation of the load identification algorithm. A python script was used to create a feature set that would be used to train and test the proposed algorithm.

4.3.4 Load Identification

It is in this subsection that the feedforward ANN is implemented. The neural network was implemented with one input node, five output nodes, eight hidden nodes at a learning rate of 0.3. Figure 4.3 shows a visual representation of the network.

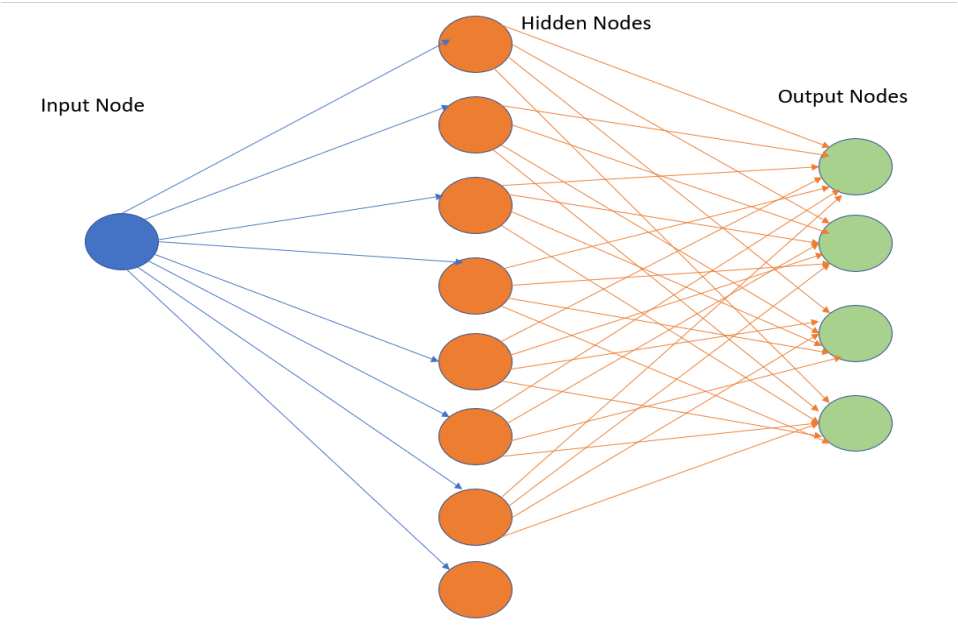


Figure 4.3 Feedforward ANN node diagram.

A python script (find attached in the appendix) converts data stored in a csv file to arrays. The code randomly runs through each item in the training data to avoid convergence on local minimums. The data is fed into the network and the activation of errors, output layer's nodes and hidden layer's nodes are calculated. The error is back propagated to the hidden layer and compared to a specified threshold. The process repeats till the error is as minimum as specified. A flow diagram below specifies how the Feedforward ANN would be implemented.

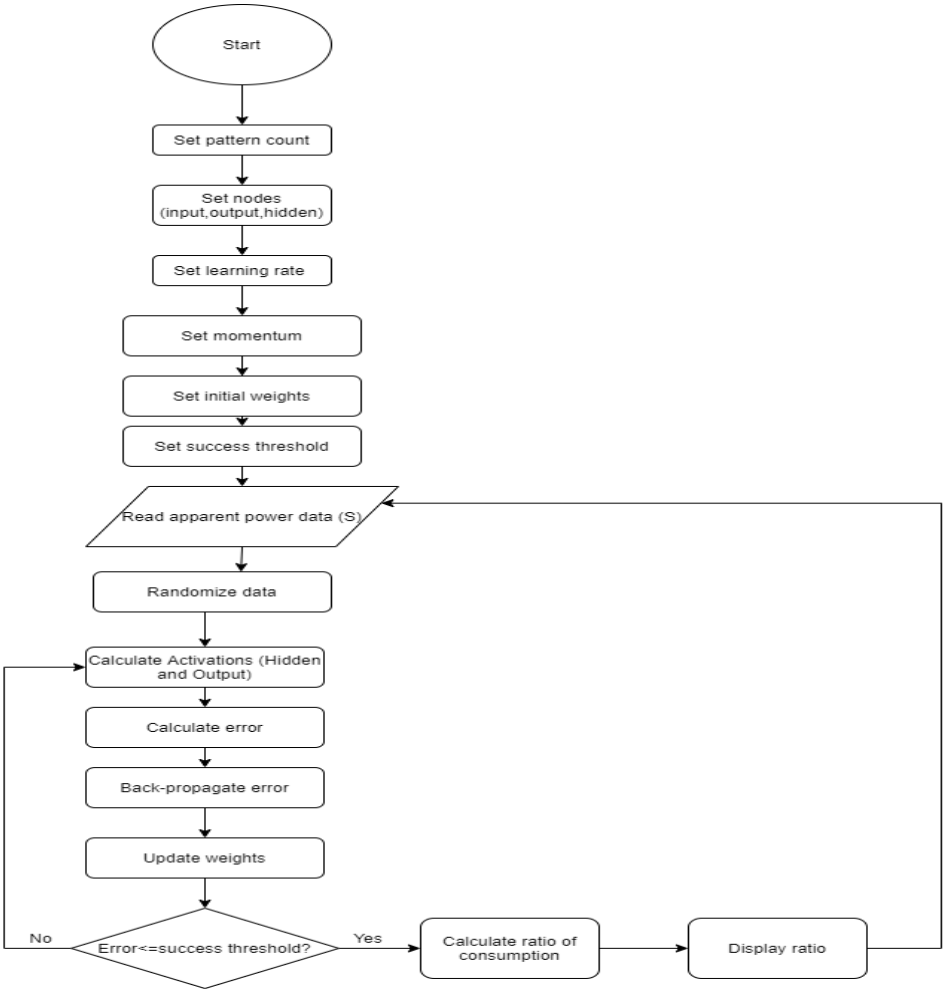


Figure 4.4 Feedforward ANN flow diagram.

4.3.5 Training and Testing the System

Based on the N-fold cross validation, the dataset obtained was divided; 70% into training data and 30% to testing data. That is, 700x6 matrix is fed into the algorithm for training in batches of 10 due to the limited memory available Arduino Uno, and 300x6 matrix is used for testing. A python script converts the format of data in the csv file to arrays that can be read in Arduino. The network is then trained, and the error of the training is recorded. Test data is then passed through the network and its output is recorded for statistical analysis.

4.3.6 Output of System

The appliances are clustered per tenant and weighted based on their apparent power consumption. The ratio of consumption is calculated using the apparent power consumption of appliances that are on per tenant. The output of the system is displayed in Arduino's serial Monitor and on an OLED screen, along with the ratio of tenant consumption. For testing, two tenants were assumed. Tenant 1 owns the microwave and the stove. Tenant 2 owns the fridge and the electric heater. Both tenants share lighting in the home.

Chapter 5: Results and Discussion

The most efficient way of measuring the performance of a classifier is by how well it is able to correctly classify appliances [17] . The method used to measure the efficiency of the Feedforward ANN implemented is the confusion matrix proposed by [27]. Assumptions made are that 1) An appliance turned on is a positive state, 2) An appliance that is off is in the negative state. Based on this, performance of the system is calculated using the formulas below, and results compared to those obtained for HMM and its variants in [17] for house 1 REDD data.

Table 5.1: Confusion matrix

	Positive Predicted	Negative Predicted
Actual Positive	True Positive (TP)	False Negative (FN)
Actual Negative	False Positive (FP)	True Negative (TN)

$$Recall = \frac{TP}{TP+FN} \quad \textbf{Eqn 5.1}$$

$$Precision = \frac{TP}{TP+FP} \quad \textbf{Eqn 5.2}$$

$$Accuracy = \frac{TP+TN}{(TP+FP)+(FN+TN)} \quad \textbf{Eqn 5.3}$$

$$F1\ score = 2 * \frac{Precision*Recall}{(Precision+Recall)} \quad \textbf{Eqn 5.4}$$

High recall and high precision imply respectively that the classifications are well done and the probability of them being correct is also high. The testing data represented in Figure 5.1 shows that 38.51% True Positives, 51.11% True Negatives, 3.00% False Positives and 7.41% False Negatives were recorded.

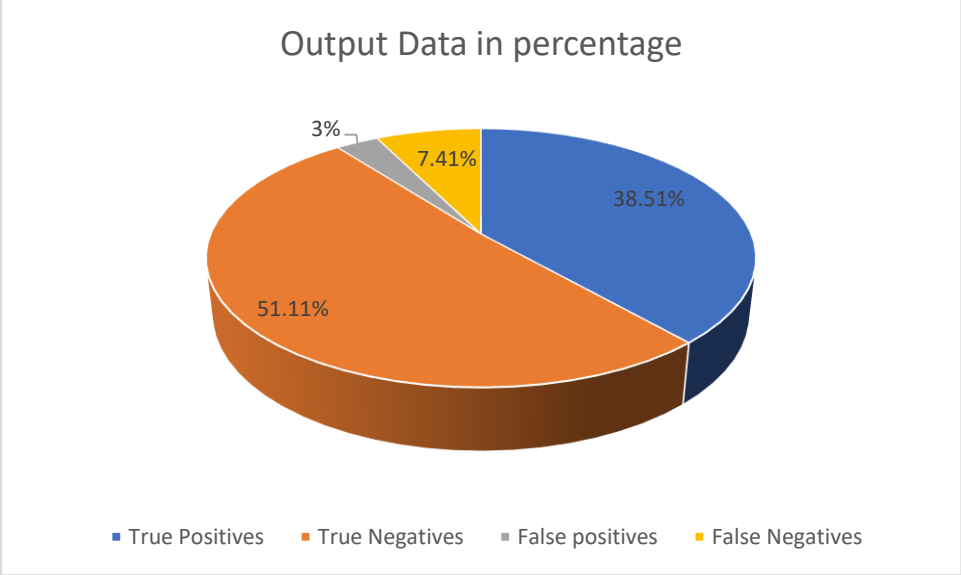


Figure 5.1 Output Data in percentage

The table below summarizes performance criteria (F1 Score) for variants of HMM and the proposed algorithm.

Table 5.2: Summary of performance of various algorithms

FHMM	Additive FHMM	Hie FHMM	Proposed Feedforward ANN
0.450	0.749	0.854	0.881

The F1 score denotes the harmonic average of recall and precision. It is considered a more general measure of overall performance. From table 5.2, it is seen that the proposed Feedforward ANN algorithm performs better than HMM and its variants. With a recall of 0.839, precision of 0.928 and an accuracy of 0.8963, the proposed ANN algorithm can efficiently predict device status. This analysis proves that Feedforward ANN is better performing than HMM and its variants.

Comparison of output from the algorithm and target data is displayed in the graph below (Figure 5.2). For the refrigerator, the algorithm data differed by 0.0016. Lighting values varied by 0.0018, and microwave data varied by 0.00178. The two devices that changed state in the timestamp graphed below were the electric heater and electric stove. The electric heater data varied by 0.0278 for OFF status and 0.0068 for ON status. The electric stove data varied by 0.0492 for OFF status and 0.0021 for ON status. It is seen that the algorithm has better estimations for ON status than OFF status. This tallies with the percentages of false negatives and false positives that were obtained. More false negatives were recorded than false positives. This is mainly based on the testing data that was available. The false data obtained also occurred mainly in the training cycles that contained some devices switching states from ON to OFF and vice versa. These false data are mainly due to the microprocessor available. A microcontroller, such as the ATmega2560 would have reduced the occurrence of false data since larger datasets can be used in training at a time. Larger datasets give the algorithm more samples to train with at a given time.

Feedforward ANN, though efficient in appliance detecting, would be efficient in giving only a general idea of the amount of each tenant consumes. That is to say, the estimations given here would be effective only for splitting the bill, and not for calculating the exact amount each tenant has consumed. This was the focus based on the requirements stated in the interviews discussed. The various appliances are clustered based on their owners, and the ratio of consumption is calculated.

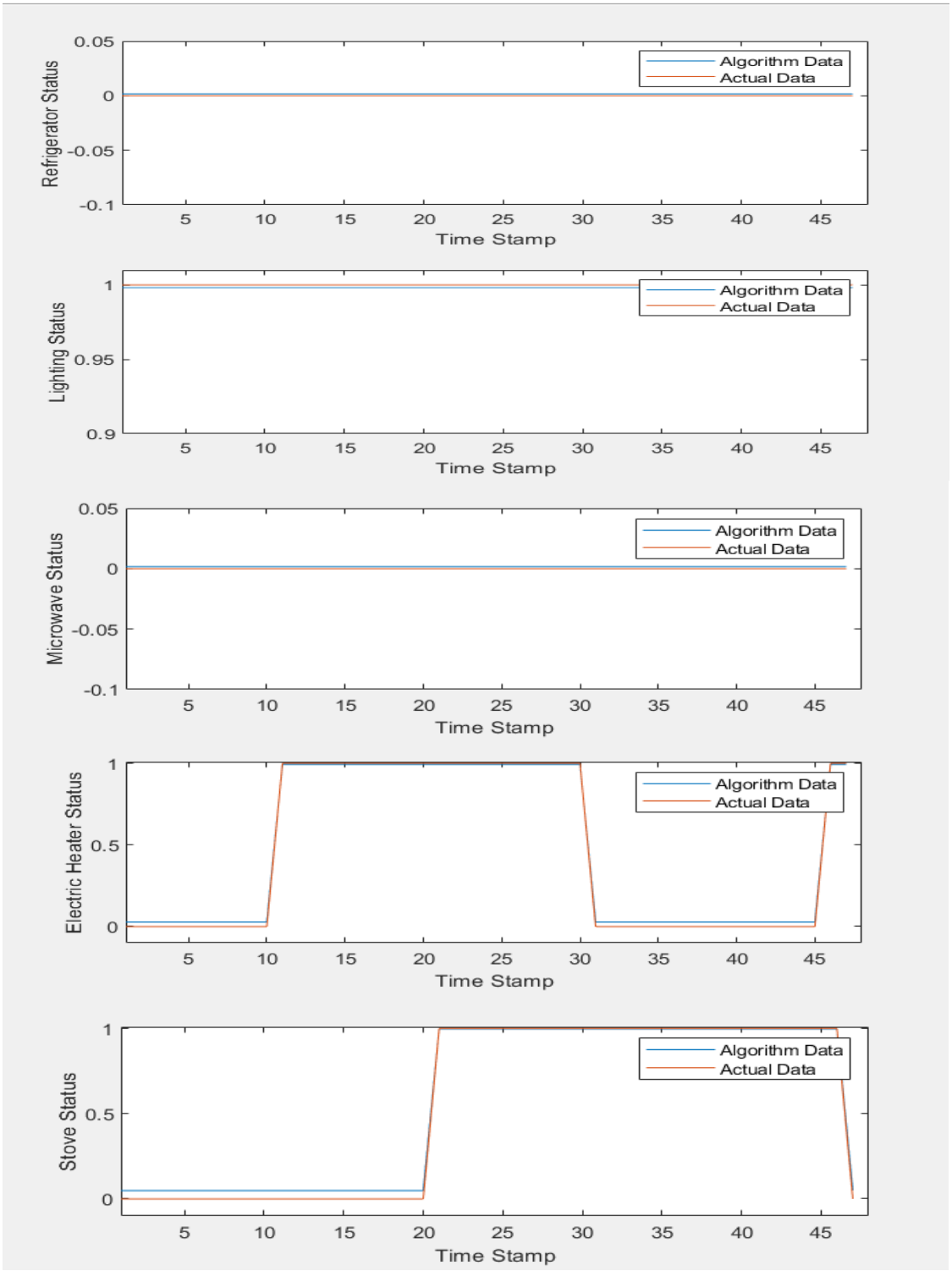


Figure 5.2 Algorithm data compared to target data.

The Feedforward ANN algorithm requires initial assessment of appliances in each tenant's possession. It would depend on the tenants fully co-operating and providing credible information, with the understanding that the system is to provide an objective method of splitting bills, which is advantageous to them. Per the system setup now, should a tenant

purchase a new appliance, the system would need to be inputted into the system, and the system trained to detect this appliance as well. Possible ways of improving this are discussed in Chapter 6.

There is the possibility that tenants may have devices with similar apparent power consumption. In this system, devices are clustered per tenant for consumption calculations. As such, even if devices owned by different tenants have similar consumptions, the consumption calculations are not affected. However, load identification may not be accurate. The use of more parameters as input nodes are ideal in differentiating such devices. Common parameters used to differentiate electrically similar devices include current harmonics and transient properties such as current spikes, spectral envelopes and transient response time [7]. However, this requires large sampling rates to identify such transient properties. This scenario was not implemented in this system because of the minimum memory and processing power of the available microcontroller, as well as the unavailability of such dataset.

Depictions of the output of the proposed system is shown in Figure 5.3. The system displays the ON and OFF state of each device every thousand cycles.

Figure 5.3 Testing of the Feedforward ANN in Arduino Displayed in Serial Monitor.

At the end of a successful prediction, the ratio of tenants' consumption is displayed in the serial monitor as shown in Figure 5.4. it is seen that Tenant 1 who owns lighting, a microwave and stove had his/her lights and electric stove on at the time stamp captured. Tenant 2 had lights and electric heater on at the time captured. The system calculates the apparent power consumption per tenant and displays it as a percentage of the bill to be paid on the OLED as shown in Figure 5.5.

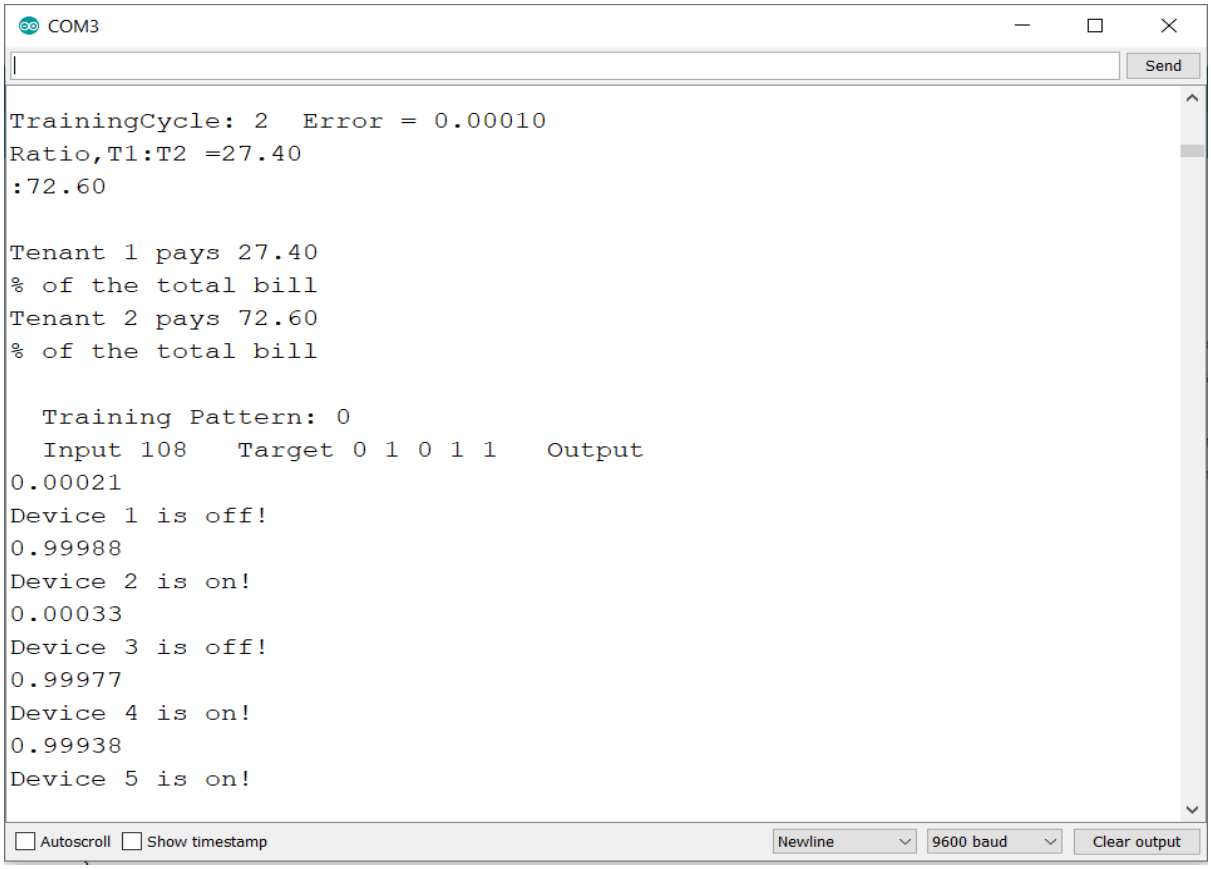


Figure 5.4 Serial port displaying ratio each tenant has to pay for electricity as at the time stamp captured.

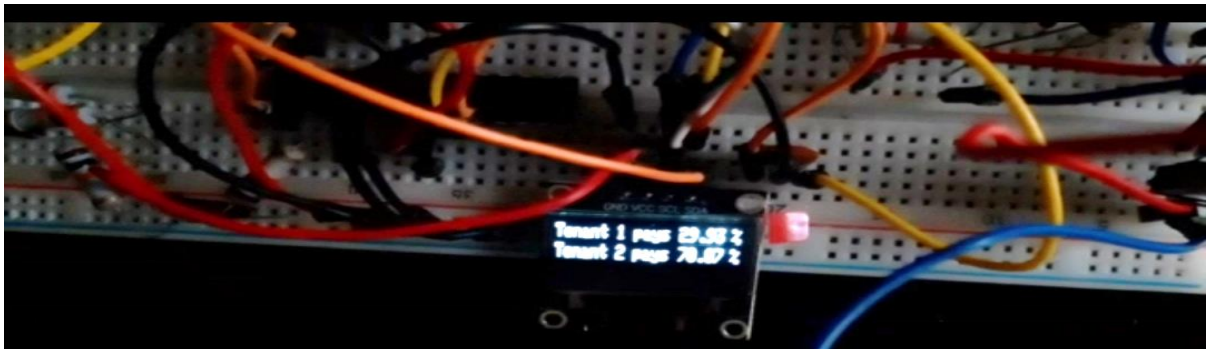


Figure 5.5 Image of Ratios displayed on OLED

Chapter 6: Conclusion

The NILM-based energy monitoring system proposed in this capstone is efficient for application in compound houses in Ghana. The deductions, limitations and further work are discussed below.

6.1 Conclusion

The NILM-Based Energy monitoring system was implemented using the Feedforward ANN in Arduino. The capstone project compared this method to the Hidden Markov Model and its variants to determine which performs better. This was done by training the proposed algorithm with House One REDD Data, narrowed down to five appliances; refrigerator, lighting, microwave, electric heater, and stove sampled at three seconds. From measuring performance criteria for HMM and its variants, it is concluded that the proposed feedforward ANN performs better in this NILM system and is appropriate for disaggregation specifically in Compound Houses.

6.2 Challenges Faced

Challenges faced during the project include the COVID-19 pandemic that hindered the purchase of components. Purchasing these hardware components would have enabled the full implementation of the approach proposed. Due to this, data of the same appliance of different makes could not be implemented to mimic an ideal compound house.

There was also a challenge comparing NILM algorithm performances implementations. That is to say that even though the NILM Toolkit provides standard data for testing various algorithms, there is little to no access to implemented NILM algorithms. This makes it difficult to determine if the comparisons made between various implementations of the same types of algorithms are indeed representative of the performance of the systems. For

instance, HMM may be implemented in different ways by different authors. Though this gives room for innovation in NILM, it poses a problem in effectively comparing algorithms. The true performance of the system cannot be based only on the algorithm but must also be based on how authors implemented these algorithms.

The available microcontroller, ATMEGA 328P could not fully support both the algorithm and an OLED display. This is because the OLED requires a minimum of 1K Static Random Access Memory (SRAM). The algorithm coupled with the OLED SRAM requirements could not be supported by the 2K SRAM of the ATMEGA 328P. A more powerful microcontroller, such as the ATmega2560 would be able to support the system.

6.3 Future Work and Recommendation

In future work, a local dataset of electrical power consumption in compound houses will be built to support further research in this area. The proposed system would also be implemented in hardware. This would present the opportunity to evaluate the performance of the entire system, and not just the software aspect. Further work also needs to be done in exploring an unsupervised algorithm as robust as the algorithm proposed in this capstone to cater for the addition of new appliances tenants may purchase. This approach would address the problem of the system not capturing new appliances purchased by a tenant. Further work would also be done to display actual fractions of electricity consumption, instead of a ratio. This would require the addition of a real time clock since power calculations would require real time. The use of a power adapter in the system would also increase the possible features that can be used in load identification.

A recommendation to the NILM body of knowledge would be that researchers make available their datasets along with their research. This would aid in effective comparison between various algorithms. The lack of the availability of datasets used in various NILM

implementations do not allow for comparative analysis to be done on different implementations of common NILM algorithms. Researchers should also provide performance criteria using the datasets available in the NILM toolkit, to provide a standard for comparison.

References

[1] Ministry of Energy,Ghana, “Ministry of Energy Sector Overview,” 2017. https://www.energymin.gov.gh/sector-overview (accessed Feb. 24, 2020).

[2] “Ghana electricity prices, March 2019 | GlobalPetrolPrices.com,” Mar. 2019. https://www.globalpetrolprices.com/Ghana/electricity_prices/ (accessed Oct. 13, 2019).

[3] C. N. D. Laryea, “An investigation into the factors accounting for the high and persistent housing deficit in Ghana,” Thesis, 2018.

[4] Ghana Statistical Service, “Ghana Statistical Service, 2010 Population and Housing Census, Summary Report of Final Results.” Accra, Ghana, 2012.

[5] R. Saaka A., “A SMART ENERGY MONITOR FOR COMPOUND HOUSES THAT USE ONE SHARED METER,” Thesis, Ashesi University, 2019.

[6] F. D. S. N. Babereyir, “The cost- effectiveness of NED providing meters to each tenant in compound houses in northern area,” Thesis, 2011.

[7] J. Revuelta Herrero *et al.*, “Non Intrusive Load Monitoring (NILM): A State of the Art,” in *Trends in Cyber-Physical Multi-Agent Systems. The PAAMS Collection - 15th International Conference, PAAMS 2017*, Cham, 2018, pp. 125–138, doi: 10.1007/978-3-319-61578-3_12.

[8] H. Pihala, “Power signatures of home appliances based on Non-Intrusive Appliance Load Monitoring (NIALM) method,” Smart Grids and Energy Markets, Finland, Research VTT-R-02647-12, 2012.

[9] G. W. Hart, “Nonintrusive appliance load monitoring,” *Proc. IEEE*, vol. 80, no. 12, pp. 1870–1891, Dec. 1992, doi: 10.1109/5.192069.

[10] C. Laughman *et al.*, “Power signature analysis,” *IEEE Power Energy Mag.*, vol. 1, no. 2, pp. 56–63, Mar. 2003, doi: 10.1109/MPAE.2003.1192027.

[11] H. Y. Lam, G. S. K. Fung, and W. K. Lee, “A Novel Method to Construct Taxonomy Electrical Appliances Based on Load Signaturesof,” *IEEE Trans. Consum. Electron.*, vol. 53, no. 2, pp. 653–660, May 2007, doi: 10.1109/TCE.2007.381742.

[12] M. Weiss, A. Helfenstein, F. Mattern, and T. Staake, “Leveraging smart meter data to recognize home appliances,” in *2012 IEEE International Conference on Pervasive Computing and Communications*, Mar. 2012, pp. 190–197, doi: 10.1109/PerCom.2012.6199866.

[13] A. Ruano, A. Hernandez, J. Ureña, M. Ruano, and J. García, “NILM Techniques for Intelligent Home Energy Management and Ambient Assisted Living: A Review,” *Energies*, vol. 12, p. 2203, Jun. 2019, doi: 10.3390/en12112203.

[14] D. Benyoucef, P. Klein, and T. Bier, “Smart Meter with non-intrusive load monitoring for use in Smart Homes,” in *2010 IEEE International Energy Conference*, Dec. 2010, pp. 96–101, doi: 10.1109/ENERGYCON.2010.5771810.

[15] M. Lu and Z. Li, “A Hybrid Event Detection Approach for Non-Intrusive Load Monitoring,” *IEEE Trans. Smart Grid*, vol. 11, no. 1, pp. 528–540, Jan. 2020, doi: 10.1109/TSG.2019.2924862.

[16] B. Zhao, K. He, L. Stankovic, and V. Stankovic, “Improving Event-Based Non-Intrusive Load Monitoring Using Graph Signal Processing,” *IEEE Access*, vol. 6, pp. 53944–53959, 2018, doi: 10.1109/ACCESS.2018.2871343.

[17] J. Kim, T.-T.-H. Le, and H. Kim, “Nonintrusive Load Monitoring Based on Advanced Deep Learning and Novel Signature,” *Comput. Intell. Neurosci.*, vol. 2017, p. 4216281, Oct. 2017, doi: 10.1155/2017/4216281.

[18] “Introduction to Current Transformers.” Elkor Technologies, Jul. 05, 2006.

[19] E. Brennan, “Electricity Guide For Students: Volts, Amps, Watts, Ohms, AC and DC,” Mar. 31, 2020. https://owlcation.com/academia/Watts-Amps-Kilowatt-Hours-What-Does-it-All-mean (accessed Apr. 19, 2020).
[20] G. Quesada, A. Conesa, R. Bargalló, and M. Roman, “Burden Resistor Selection in Current Transformers for Low Power Applications,” 2014.
[21] “CT sensors - Interfacing with an Arduino | Archived Forum.” https://openenergymonitor.org/forum-archive/node/156.html (accessed Apr. 19, 2020).
[22] FairchildSemiconductor, “LM7805 pdf, LM7805 description, LM7805 datasheets, LM7805 view ::: ALLDATASHEET :::,” 2005. https://pdf1.alldatasheet.com/datasheet-pdf/view/131011/FAIRCHILD/LM7805.html (accessed Apr. 19, 2020).
[23] “Arduino - ArduinoToBreadboard.” https://www.arduino.cc/en/Tutorial/ArduinoToBreadboard (accessed Apr. 19, 2020).
[24] Winstar, “4 pin OLED Display - Winstar Display,” 2020. https://www.winstar.com.tw/products/oled-module/graphic-oled-display/4-pin-oled.html (accessed Apr. 19, 2020).
[25] Z. Kolter J. and M. Johnson J., “REDD: A Public Data Set for Energy Disaggregation Research,” MIT, San Diego, CA ,USA, 2011.
[26] N. Batra *et al., NILMTK: An open source toolkit for non-intrusive load monitoring*. 2014.
[27] F. Provost and R. Kohavi, “Guest Editors’ Introduction: On Applied Research in Machine Learning,” *Mach. Learn.*, vol. 30, no. 2–3, pp. 127–132, 1998, doi: 10.1023/A:1007442505281.

Appendix

A1. Interview Questions and Responses

Questions prepared for the interview are listed below. However, as conversations continued, other questions were asked based on responses from tenants of compound houses. Summaries of answers obtained are recorded below.

1. Who are your electricity providers?

   Ans: All three receive power from only Electricity Company of Ghana/ Power Distribution Service.

2. Do you use the prepaid or postpaid meters?

   Ans: All three use the prepaid system.

3. What type of compound house do you live in? What resources do you share? Is electricity inclusive?

   Ans: Two of the tenants live in 'chamber and hall setting' and share electricity meters, water meters and a bathroom. The third tenant shares only electricity and water bills with other tenants.

4. If you do share meters with others, how many of you share a meter? How do you share the cost?

   Ans: One tenant shares a meter with two other households. Each of the other two of the tenants share meters with three other households. The various methods used in sharing bills are splitting it evenly, taking turns to pay each month, having it factored into their rent, and splitting the bill based on general assessment of electrical devices in the house. None of these arrangements had been effective.

5. What are the issues that arise from your various methods of sharing the bills?

   Ans: All three tenants feel cheated, especially in instances when they were hardly at home. Sometimes, other tenants do not pay their share of the bill.

6. What should a good solution to this problem do?

   Ans: In an attempt to curb the issues that arise from sharing electricity bills, one tenant bought her own meter priced over a thousand Ghana Cedis (GHS1000.00). Her electricity bill has increased as compared to previously when she shared a meter, but she mentioned that she now has her peace. The other two have done nothing to improve the situation.

   According to all three tenants, a good solution to this problem would be a cheap way of splitting the bill fairly. It should be visible by all and available at all times, so that whenever it is time to split the bill, it can be accessed.

7. How often do you pay your electricity prepaid bills?

   Ans: All three pay their bills as and when their prepaid credits get finished. As such, there is no regular interval in paying for electricity.

A2. Code for appending output values to Data Set. Implemented in Python

```
import csv


'''
The function append_output goes through the nilm dataset and checks
to see if
a particular device is on or off. If a device is 'on', the device is
represented
with a 1 in the dataset, however if a device is 'off', the device is
represented
with a 0 in the dataset.
'''
def append_output(csvfile,writtenfile):
      final_file = []
      #open 'nilm_data.csv' and read data from that file
      csv_data_file = open(csvfile,'r')
      #open the 'outputfile.csv' and write the data into that file
      file_reader = csv.reader(csv_data_file)
      #for each row of data in the 'nilm_data.csv', check to see if
each device is 'on' or 'off'.
      #if the device is 'on' append a '1' to the row of data, however
if the device is 'off' append
      #a '0' to the row of data.
      for row in file_reader:
            new_row = []
            for j in range(len(row)):
                  new_row.append(row[j])
            for i in range(len(row)):
```

```
                if row[i] != '':
                        try:
                                if int(row[i]) <= 7:
                                        new_row.append(0)
                                else:
                                        new_row.append(1)
                        except Exception as e:
                                if float(row[i]) <= 7:
                                        new_row.append(0)
                                else:
                                        new_row.append(1)

                else:
                        continue

        empty_str = ''
        for j in new_row:
                empty_str = empty_str + ','+ str(j)
        final_string = empty_str.lstrip(',')
        final_file.append(final_string)

    #print out the rows of data before it is written into the
'outputfile.csv'
    results_file_csvfile = open(writtenfile, 'a')
    for i in final_file:
        line = ''
        line = line + i + '\n'
        print(line)
```

```
            results_file_csvfile.write(line)
      results_file_csvfile.close()


append_output('nilm_data_use.csv','outputfile.csv')
```

A3. Code for separating data into input nodes and output nodes. The python script saves the data in Arduino- compatible arrays.

```
def generate_set(csvfile, writtenInputFile, writtenOutputFile):
      #read a row of data
      #take out all of the input data, write it in a string
      #place {} around that string , place that string in an
      #input array. Do the same thing for the output data.
      #Place the output data into an output array.
      #Write the results of both arrays into files.
      final_input_file = []
      final_output_file = []
      with open(csvfile) as text_file:
          file_reader = csv.reader(text_file, delimiter=',')
          for row in file_reader:
            set_input_string = ''
            set_output_string = ''
            for i in range(0,1):
                  set_input_string = set_input_string + ',' + row[i]
            set_input_string = set_input_string.lstrip(',')
            set_input_string = '{'+set_input_string+'}'+','
            final_input_file.append(set_input_string)
            for j in range(6,11):
                  set_output_string = set_output_string + ',' + row[j]
            set_output_string = set_output_string.lstrip(',')
            set_output_string = '{'+set_output_string+'}'+','
            final_output_file.append(set_output_string)
      print("input dataset: ")
      print()
```

```
    results_file_csvfile = open(writtenInputFile, 'a')
    for i in final_input_file:
        print(i)
        results_file_csvfile.write(i+ '\n')
    print('_____________________________________')
    print()
    print("output dataset: ")
    print()
    results_file_csvfile.close()
    output_csv_file = open(writtenOutputFile, 'a')
    for i in final_output_file:
        print(i)
        output_csv_file.write(i+ '\n')
    output_csv_file.close()



generate_set('nilm_train.csv','feature_data.csv','output_data.csv')
```

A4. Python Code for splitting data into training set and testing set

```
import csv


'''
The split_data function splits the 'outputfile.csv' into training and
testing data

'''
def split_data(inputfile,trainfile,testfile):
      inputfile_data = open(inputfile,'r')
      traindata_file = open(trainfile, 'a')
      testdata_file = open(testfile, 'a')
      inputfile_reader = csv.reader(inputfile_data)
      count = 1
      for row in inputfile_reader:
            if count <= 700:
                  line = ''
                  for i in range(len(row)):
                        line = line + ','+ row[i]
                  line  = line.lstrip(',')
                  line = line + '\n'
                  traindata_file.write(line)
                  count += 1
            elif count > 700 and count < 1001:
                  line = ''
                  for i in range(len(row)):
                        line = line + ','+ row[i]
```

```
                line  = line.lstrip(',')
                line = line + '\n'
                testdata_file.write(line)
          else:
                continue
     traindata_file.close()
     testdata_file.close()


split_data('outputfile.csv','nilm_train.csv','nilm_test.csv')
```

A5. Arduino code for Feedforward ANN and splitting of bill

```
//Inspired by the structure of ANN implemented by Ralph Heymsfeld
///Arduino Uno is used in this project mainly because of its
availability.
//To be able to train using more data at a time, an arduino mega would
be ideal as it SRAM is more than 2k



#include <math.h>
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#define OLED_RESET 4
Adafruit_SSD1306 display(OLED_RESET);



/**************************************************************
 * Network Configuration - customized per network
 **************************************************************/

const int PatternCount = 10; //The number of rows of the training data
const int InputNodes = 1; //Number of associated neurons for the input
const int HiddenNodes = 8; //Number of neurons associated with the
hidden layer.They need to be at least the number of output neurons.
const int OutputNodes = 5; //Number of neurons associated with the
output layer
```

```
const float LearningRate = 0.3;// The proportion of the error that is back-propagated. This value is chosen to prevent the network from going into oscillation
const float Momentum = 0.9;//The proportion of the previous iteration that affects current iteration
const float InitialWeightMax = 0.5;//Maximum starting value for the randomly assigned weights.
const float Success = 0.0004;//Threshold for measuring successful training
const float weight_dev[1][5]={430,82,1550,1220,400}; //Apparent power of each device

const byte Input[PatternCount][InputNodes] = {
//The vector for the apparent data goes here.

};

const byte Target[PatternCount][OutputNodes] = {
//The output matrix goes here

};

/************************************************************
 * End Network Configuration
 ************************************************************/


int i, j, p, q, r,l;
```

```
int ReportEvery1000;
int RandomizedIndex[PatternCount];
long  TrainingCycle;
float Rando;
float Error;
float Accum;



float Hidden[HiddenNodes];
float Output[OutputNodes];
float HiddenWeights[InputNodes+1][HiddenNodes];
float OutputWeights[HiddenNodes+1][OutputNodes];
float HiddenDelta[HiddenNodes];
float OutputDelta[OutputNodes];
float ChangeHiddenWeights[InputNodes+1][HiddenNodes];
float ChangeOutputWeights[HiddenNodes+1][OutputNodes];
float tenant1_ratio;
float tenant2_ratio;



void setup(){
  Serial.begin(9600);
  randomSeed(analogRead(3));
  ReportEvery1000 = 1;
  for( p = 0 ; p < PatternCount ; p++ ) {
    RandomizedIndex[p] = p ;
     }
```

```
    if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) { // Address 0x3D
for 128x64
    Serial.println(F("SSD1306 allocation failed"));
    display.display();
    display.clearDisplay();
  }



}

void loop (){


/******************************************************************
* Initialize HiddenWeights and ChangeHiddenWeights
******************************************************************/

  for( i = 0 ; i < HiddenNodes ; i++ ) {
    for( j = 0 ; j <= InputNodes ; j++ ) {
      ChangeHiddenWeights[j][i] = 0.0 ;
      Rando = float(random(100))/100;
      HiddenWeights[j][i] = 2.0 * ( Rando - 0.5 ) * InitialWeightMax
;
    }
  }
/******************************************************************
* Initialize OutputWeights and ChangeOutputWeights
******************************************************************/
```

```
  for( i = 0 ; i < OutputNodes ; i ++ ) {
    for( j = 0 ; j <= HiddenNodes ; j++ ) {
      ChangeOutputWeights[j][i] = 0.0 ;
      Rando = float(random(100))/100;
      OutputWeights[j][i] = 2.0 * ( Rando - 0.5 ) * InitialWeightMax
;
    }
  }
  //Serial.println("Initial/Untrained Outputs: ");
  toTerminal();
/******************************************************************
* Begin training
******************************************************************/

  for( TrainingCycle = 1 ; TrainingCycle < 9483647 ; TrainingCycle++)
{

/******************************************************************
* Randomize order of training patterns
******************************************************************/

    for( p = 0 ; p < PatternCount ; p++) {
      q = random(PatternCount);
      r = RandomizedIndex[p] ;
      RandomizedIndex[p] = RandomizedIndex[q] ;
      RandomizedIndex[q] = r ;
    }
```

```
    Error = 0.0 ;
/*****************************************************************
* Cycle through each training pattern in the randomized order
*****************************************************************/
    for( q = 0 ; q < PatternCount ; q++ ) {
      p = RandomizedIndex[q];

/*****************************************************************
* Compute hidden layer activations
*****************************************************************/

      for( i = 0 ; i < HiddenNodes ; i++ ) {
        Accum = HiddenWeights[InputNodes][i] ;
        for( j = 0 ; j < InputNodes ; j++ ) {
          Accum += Input[p][j] * HiddenWeights[j][i] ;
        }
        Hidden[i] = 1.0/(1.0 + exp(-Accum)) ;
      }

/*****************************************************************
* Compute output layer activations and calculate errors
*****************************************************************/

      for( i = 0 ; i < OutputNodes ; i++ ) {
        Accum = OutputWeights[HiddenNodes][i] ;
        for( j = 0 ; j < HiddenNodes ; j++ ) {
          Accum += Hidden[j] * OutputWeights[j][i] ;
        }
```

```
        Output[i] = 1.0/(1.0 + exp(-Accum)) ;   //logistic function.
Useful in back propagation because it is a continuous derivative.
        OutputDelta[i] = (Target[p][i] - Output[i]) * Output[i] * (1.0
- Output[i]) ;//Calculation of errors(actual-target)
        Error += 0.5 * (Target[p][i] - Output[i]) * (Target[p][i] -
Output[i]) ;
      }

/****************************************************************
* Backpropagate errors to hidden layer
****************************************************************/

      for( i = 0 ; i < HiddenNodes ; i++ ) {
        Accum = 0.0 ;
        for( j = 0 ; j < OutputNodes ; j++ ) {
          Accum += OutputWeights[i][j] * OutputDelta[j] ;
        }
        HiddenDelta[i] = Accum * Hidden[i] * (1.0 - Hidden[i]) ;
      }

/****************************************************************
* Update Inner-->Hidden Weights
****************************************************************/


      for( i = 0 ; i < HiddenNodes ; i++ ) {
        ChangeHiddenWeights[InputNodes][i]    =    LearningRate    *
HiddenDelta[i] + Momentum * ChangeHiddenWeights[InputNodes][i] ;
```

```
        HiddenWeights[InputNodes][i]                                    +=
ChangeHiddenWeights[InputNodes][i] ;
        for( j = 0 ; j < InputNodes ; j++ ) {
          ChangeHiddenWeights[j][i] = LearningRate * Input[p][j] *
HiddenDelta[i] + Momentum * ChangeHiddenWeights[j][i];
          HiddenWeights[j][i] += ChangeHiddenWeights[j][i] ;//Weights
between the input an dhidden layer
        }
      }

/*****************************************************************
* Update Hidden-->Output Weights
*****************************************************************/

      for( i = 0 ; i < OutputNodes ; i ++ ) {
        ChangeOutputWeights[HiddenNodes][i]    =    LearningRate    *
OutputDelta[i] + Momentum * ChangeOutputWeights[HiddenNodes][i] ;
        OutputWeights[HiddenNodes][i]                                   +=
ChangeOutputWeights[HiddenNodes][i] ;
        for( j = 0 ; j < HiddenNodes ; j++ ) {
          ChangeOutputWeights[j][i] = LearningRate * Hidden[j] *
OutputDelta[i] + Momentum * ChangeOutputWeights[j][i] ;
          OutputWeights[j][i] += ChangeOutputWeights[j][i] ;//Weights
between the hidden layer and the output.
        }
      }
    }
```

```
/*****************************************************************
* Every 1000 cycles send data to terminal for display
*****************************************************************/
    ReportEvery1000 = ReportEvery1000 - 1;
    if (ReportEvery1000 == 0)
    {
      //Serial.println();
      //Serial.println();
      //Serial.print ("TrainingCycle: ");
      //Serial.print (TrainingCycle);
      //Serial.print ("  Error = ");
      //Serial.println (Error, 5);
      //Calculate the ratio. Appliances belonging to Tenant 1 are
lighting, microwave  and stove  . Tenant 2 owns lighting,fridge and
heating
      //const float weight_dev[1][5] is in the order friDge  light
microwave electric heater stove
 /*****************************************************************
* Calculate ratio of each tenant's consumption
*****************************************************************/
      tenant1_ratio=
tenant1_ratio+(weight_dev[0][1])*(Output[1])+((weight_dev[0][2])*Out
put[2])+((weight_dev[0][4])*Output[4]);
      tenant2_ratio=
tenant2_ratio+(weight_dev[0][0])*(Output[0])+((weight_dev[0][1])*Out
put[1])+((weight_dev[0][3])*Output[3]);

      toTerminal();
```

```
      if (TrainingCycle==1)
      {
        ReportEvery1000 = 999;
      }
      else
      {
        ReportEvery1000 = 1000;
      }
    }
/*****************************************************************
* If error rate is less than pre-determined threshold then end
*****************************************************************/
        if( Error < Success ) break ;
  }

/*****************************************************************
* Display Ratio So Far ON OLED
*****************************************************************/
  display.setTextSize(1);
  display.setTextColor(WHITE);
  display.clearDisplay();
  display.setCursor(0,0);
  display.println("Tenant 1 pays");
  display.setCursor(85,0);

display.println((tenant1_ratio/(tenant1_ratio+tenant2_ratio))*100);
  display.setCursor(120,0);
```

```
  display.println("%");
  display.setCursor(0,10);
  display.println("Tenant 2 pays");
  display.setCursor(85,10);

display.println((tenant2_ratio/(tenant1_ratio+tenant2_ratio))*100);
  display.setCursor(120,10);
  display.println("%");
  display.display();
/*****************************************************************
*Print Output To Serial Monitor for Monitoring
*****************************************************************/
  Serial.println ();
  Serial.println();
  Serial.print ("TrainingCycle: ");
  Serial.print (TrainingCycle);
  Serial.print ("  Error = ");
  Serial.println (Error, 5);
  Serial.print ("Ratio,T1:T2 =");
  Serial.println
((tenant1_ratio/(tenant1_ratio+tenant2_ratio))*100);
  Serial.print (":");
  Serial.println
((tenant2_ratio/(tenant1_ratio+tenant2_ratio))*100);
  Serial.print ("\n");
  Serial.print ("Tenant 1 pays ");
  Serial.println
((tenant1_ratio/(tenant1_ratio+tenant2_ratio))*100);
```

```
  Serial.print ("% of the total bill \n");
  Serial.print ("Tenant 2 pays ");
  Serial.println
((tenant2_ratio/(tenant1_ratio+tenant2_ratio))*100);
  Serial.print ("% of the total bill\n");

  toTerminal();

  Serial.println ();
  Serial.println ();
  Serial.println ("Training Set Solved! ");
  Serial.println ("--------");
  Serial.println ();
  Serial.println ();

  ReportEvery1000 = 1;

 }


void toTerminal()
{

  for( p = 0 ; p < PatternCount ; p++ ) {
    Serial.println();
    Serial.print ("  Training Pattern: ");
    Serial.println (p);
    Serial.print ("  Input ");
```

```
    for( i = 0 ; i < InputNodes ; i++ ) {
      Serial.print (Input[p][i], DEC);
      Serial.print (" ");
    }
    Serial.print ("  Target ");
    for( i = 0 ; i < OutputNodes ; i++ ) {
      Serial.print (Target[p][i], DEC);
      Serial.print (" ");
    }
/******************************************************************
* Compute hidden layer activations
******************************************************************/

    for( i = 0 ; i < HiddenNodes ; i++ ) {
      Accum = HiddenWeights[InputNodes][i] ;
      for( j = 0 ; j < InputNodes ; j++ ) {
        Accum += Input[p][j] * HiddenWeights[j][i] ;
      }
      Hidden[i] = 1.0/(1.0 + exp(-Accum)) ;
    }

/******************************************************************
* Compute output layer activations and calculate errors
******************************************************************/

    for( i = 0 ; i < OutputNodes ; i++ ) {
      Accum = OutputWeights[HiddenNodes][i] ;
      for( j = 0 ; j < HiddenNodes ; j++ ) {
```

```
      Accum += Hidden[j] * OutputWeights[j][i] ;
    }
    Output[i] = 1.0/(1.0 + exp(-Accum)) ;
  }
  //Serial.print ("  Output \n");
  for( i = 0 ; i < OutputNodes ; i++ ) {

    Serial.print (Output[i], 5);
    Serial.print ("\n");
    if( Output[i] < 0.5 ) {
        Serial.print ("Device ");
        Serial.print (i+1);
        Serial.print (" is off!");
        Serial.print ("\n");
      }
      else{
        Serial.print ("Device ");
        Serial.print (i+1);
        Serial.print (" is on!");
        Serial.print ("\n");

}
}
}
}
```